

# Technical Report: NUS-ACT-11-001-Ver.2:

## Fault-tolerant Cooperative Tasking for Multi-agent Systems

### Preprint, Submitted for publication

Mohammad Karimadini, and Hai Lin,

#### Abstract

A natural way for cooperative tasking in multi-agent systems is through a top-down design by decomposing a global task into subtasks for each individual agent such that the accomplishments of these subtasks will guarantee the achievement of the global task. In our previous works [1], [2], we presented necessary and sufficient conditions on the decomposability of a global task automaton between cooperative agents. As a follow-up work, this paper deals with the robustness issues of the proposed top-down design approach with respect to event failures in the multi-agent systems. The main concern under event failure is whether a previously decomposable task can still be achieved collectively by the agents, and if not, we would like to investigate that under what conditions the global task could be robustly accomplished. This is actually the fault-tolerance issue of the top-down design, and the results provide designers with hints on which events are fragile with respect to failures, and whether redundancies are needed. The main objective of this paper is to identify necessary and sufficient conditions on failed events under which a decomposable global task can still be achieved successfully. For such a purpose, a notion called passivity is introduced to characterize the type of event failures. The passivity is found to reflect the redundancy of communication links over shared events, based on which necessary and sufficient conditions for the reliability of cooperative tasking under event failures are derived, followed by illustrative examples and remarks for the derived conditions.

## I. INTRODUCTION

Multi-agent system has emerged as a hot research area with strong support from a wide range of applications such as power grids, transportation networks, ubiquitous computation, and multi-robot systems [3], [4], [5]. The significance of multi-agent systems roots in the power of parallelism and cooperation between simple components that lead to sophisticated capabilities and more robustness and functionalities than individual multi-skilled agents [6], [7], [8]. One of the key problems in multi-agent systems is top-down cooperative tasking, through which a global task is decomposed into subtasks for each individual agent such that the accomplishments of these subtasks will guarantee the achievement of the global task.

For such a purpose, in our previous work [1] a top-down design approach for multi-agent cooperative tasking was proposed for two agents and then generalized in [2] into an arbitrary finite number of agents. As the main contribution, [1], [2] identified necessary and sufficient conditions under which a deterministic task automaton is decomposable with respect to parallel composition and natural projections into local event sets, namely, the task automaton is bisimilar to the parallel composition of its natural projections. Moreover, it has been shown that if the task automaton is decomposable and local supervisors are designed to satisfy local specification automata, then the entire closed loop system satisfies the original global specification. It is worth noting here that the determinism of global task automaton does not reduce its decomposability in the sense of bisimulation into its decomposability in the sense of language equivalence [9], nor into separability of its language [10], since in general local task automata obtained by natural projection could be nondeterministic, in general (see Example 9 in the Appendix).

Once a multi-agent system is designed, its safety becomes a crucial property across the agents in order to prevent the uncompensable consequences for the system and users. Failures on the other hand are usually unavoidable due to the large scale nature and complex interactions among the distributed agents. It is therefore very important to introduce some degree of redundancy into the design so as to achieve fault-tolerance. Towards this end, this paper represents a continuation of the works in [1], [2], and deals with the robustness issues of the proposed top-down design approach with respect to event failures in the multi-agent systems. The main concern under failure is whether a previously decomposable task still can be achieved collectively by the agents. Please note that no global information on failures is assumed, and each agent is only aware of failures

around itself and just trying to accomplish its previously assigned subtask (assume that the global task is decomposable before failures, and subtasks are obtained, accordingly). An interesting question is whether these agents can achieve the original global task in spite of event failures. If not, we would like to ask under what conditions the global task could be robustly accomplished. This is actually the fault-tolerance issue of the top-down design, and the results provide designers hints on which events are fragile with respect to failures, and whether redundancies are needed for sharing of some events. It is desired to share as few number of events as possible through the communication links to reduce the bandwidth, and hence, the cost of the design. The main objective of this paper is to identify necessary and sufficient conditions on failed events under which a decomposable global task can still be achieved successfully between cooperative agents.

This work differs from diagnosability and isolation problems [11] whose interest is on detection and identification of the type of faults. In this work the faults are known and the question is the tolerance of systems in spite of the faults. It also differs from reliable supervisory control [12], [13] that seeks the minimal number of supervisors required for correct functionality of the supervised systems. Another different problem is robust supervisory control [14] that considers the plant as a set of possible plants and designs supervisor applicable for the whole range of plants.

This work is related to the fault-tolerant supervisory control that has been widely studied in the context of discrete event systems. For examples, [15] proposed switching to another supervisor after fault detection. In another work, [16], the author proposed to re-synthesis the supervisor upon the fault occurrence. A framework for fault-tolerant supervisory control has been proposed in [17] and further explored in [18] by enforcing given specifications for non-faulty and faulty parts of the plant to ensure that the plant recovers from any fault within a bounded delay, such that the recovered plant is equivalent to the non-faulty plant. In [19] a fault is modeled as an uncontrollable event, that its occurrence causes a faulty behavior. They provided a necessary and sufficient condition for the existence of supervisor under failures, based on controllability, observability and relative-closure, together with the notions of state-stability [20], [21], and language-stability [22], [23]. In [24], a fault recovery result has been proposed by introducing normal, transient and recovery modes, such that the language of the closed loop systems is equal to a given language of the normal mode. Most of these works however address the language specifications and deal with decentralized supervisory control with distributed supervisor and

monolithic plant.

In this paper, continuing the works in [1], [2], it is firstly observed that a necessary condition for preserving the decomposability is that the failed events can only be shared events and could be those that are only received from the other agents or sent to others, redundantly. In other words, a necessary condition for failed events is that they are not produced by the sensors/actuators of the corresponding agent, and that the failed events are not sent to other agents, unless there exist some alternative agents to relay them. We will call these events as passive events in the agent. Passive events indeed refer to the shared events through redundant communication links. Based on this notation, it seems that the failure of passive events have no effect on decomposability, as they do not fail in the sender agents and the receiver is just no longer informed about those events. However, it will be shown that although passivity of failure events is a necessary condition for preserving the decomposability, some additional conditions are required for the task automaton to remain decomposable. The intuitive reason is that when a shared event fails, the corresponding agent can no longer use its information as a part of decision making on the order or switch between transitions. Moreover, the failure should satisfy some criteria to ensure that after the failures, the parallel composition of local automata neither generates a new string that is not allowed in the global automaton, nor prevents a string that is allowed in the global task automaton. In particular, while the passivity of failed events is a necessary condition to preserve the decomposability, it is shown that for a deterministic task automaton that experiences failures on passive events, the task automaton remains decomposable if and only if any required decisions on order/switch between any pair of events can be accomplished by at least one of the agents after failure; no illegal string is allowed and no legal string is prevented by the composition of local task automata, after the failure. This work generalizes the preliminary work on task decomposition under failure in [25], from two agents into an arbitrary finite number of agents, providing the proofs and illustrative examples. It furthermore shows that under the passivity of failed events together with the proposed conditions, a previously achieved task automaton can be still achieved by the team of agents.

The rest of the paper is organized as follows. Section II provides preliminary lemmas, notations, definitions and recalls the necessary and sufficient conditions on decomposition of an automaton with respect to parallel composition and local event sets. The fault-tolerant task decomposability and multi-tasking problems are formulated in Section III. Sections IV presents the

main result on decomposability under event failures and introduces the necessary and sufficient conditions under which a decomposable task automaton remains decomposable in spite of event failures, followed by illustrative examples for each condition. Next, it is shown in Sections V that under passivity and the proposed conditions, if a previously decomposable task automaton has been achieved globally by local controllers, it will remain satisfied, in spite of event failures. As a special case, Section VI provides more insight on global decision making on selections and orders of transitions, in a two-agent case. Finally, the paper concludes with remarks and discussions in Section VII. The proofs of lemmas are given in the Appendix.

## II. PRELIMINARIES

The proposed top-down approach in [1], [2] investigated the deterministic global task automata and introduced necessary and sufficient conditions under which the task automaton is decomposable with respect to parallel composition and natural projections into local event sets, such that the parallel composition of local task automata bisimulates the global task automaton. It was also shown that fulfilment of local task automata, leads to satisfaction (in the sense of bisimulation) of the global task automaton. We then first recall the definition of an automaton [26].

*Definition 1: (Automaton)* A deterministic automaton is a tuple  $A := (Q, q_0, E, \delta)$  consisting of a set of states  $Q$ ; an initial state  $q_0 \in Q$ ; a set of events  $E$  that causes transitions between the states, and a transition relation  $\delta \subseteq Q \times E \times Q$ , with partial map  $\delta : Q \times E \rightarrow Q$ , such that  $(q, e, q') \in \delta$  if and only if state  $q$  is transited to state  $q'$  by event  $e$ , denoted by  $q \xrightarrow{e} q'$  (or  $\delta(q, e) = q'$ ). A nondeterministic automaton is a tuple  $A := (Q, q_0, E, \delta)$  with a partial transition map  $\delta : Q \times E \rightarrow 2^Q$ , and if hidden transitions ( $\varepsilon$ -moves) are also possible, then a nondeterministic automaton with hidden moves is defined as  $A := (Q, q_0, E \cup \{\varepsilon\}, \delta)$  with a partial map  $\delta : Q \times (E \cup \{\varepsilon\}) \rightarrow 2^Q$ . For a nondeterministic automaton the initial state can be generally from a set  $Q_0 \subseteq Q$ . Given a nondeterministic automaton  $A$ , with hidden moves, the  $\varepsilon$ -closure of  $q \in Q$ , denoted by  $\varepsilon_A^*(q) \subseteq Q$ , is recursively defined as:  $q \in \varepsilon_A^*(q)$ ;  $q' \in \varepsilon_A^*(q) \Rightarrow \delta(q', \varepsilon) \subseteq \varepsilon_A^*(q)$ . The transition relation can be extended to a finite string of events,  $s \in E^*$ , where  $E^*$  stands for *Kleene – Closure* of  $E$  (the set of all finite strings over elements of  $E$ ). For an automaton without hidden moves,  $\varepsilon_A^*(q) = \{q\}$ , and the transition on string is inductively defined as  $\delta(q, \varepsilon) = q$  (empty move or silent transition), and  $\delta(q, se) = \delta(\delta(q, s), e)$

for  $s \in E^*$  and  $e \in E$ . For an automaton  $A$ , with hidden moves, the extension of transition relation on string, denoted by  $\delta : Q \times E^* \rightarrow 2^Q$ , is inductively defined as:  $\forall q \in Q, s \in E^*, e \in E$ :  $\delta(q, \varepsilon) := \varepsilon_A^*(q)$  and  $\delta(q, se) = \varepsilon_A^*(\delta(q, s), e) = \left[ \bigcup_{q' \in \delta(q, s)} \left\{ \bigcup_{q'' \in \delta(q', e)} \varepsilon_A^*(q'') \right\} \right]$ .

The operator  $Ac(\cdot)$  [27] is then defined by excluding the states and their attached transitions that are not reachable from the initial state as  $Ac(A) = (Q_{ac}, q_0, E, \delta_{ac})$  with  $Q_{ac} = \{q \in Q \mid \exists s \in E^*, q \in \delta(q_0, s)\}$  and  $\delta_{ac} = \delta|_{Q_{ac} \times E \rightarrow Q_{ac}}$ , restricting  $\delta$  to the smaller domain of  $Q_{ac}$ . Since  $Ac(\cdot)$  has no effect on the behavior of the automaton, from now on we take  $A = Ac(A)$ .

We focus on deterministic global task automata that are simpler to be characterized, and cover a wide class of specifications. The qualitative behavior of a deterministic system is described by the set of all possible sequences of events starting from the initial state. Each such a sequence is called a string, and the collection of strings represents the language generated by the automaton, denoted by  $L(A)$ . The existence of a transition over a string  $s \in E^*$  from a state  $q \in Q$  is denoted by  $\delta(q, s)!$ . Considering a language  $L$ , by  $\delta(q, L)!$  we mean that  $\forall \omega \in L : \delta(q, \omega)!$ .

To compare the task automaton and its decomposed automata, we use the bisimulation relations [27].

*Definition 2: (Simulation and Bisimulation)* Consider two automata  $A_i = (Q_i, q_i^0, E, \delta_i)$ ,  $i = 1, 2$ . A relation  $R \subseteq Q_1 \times Q_2$  is said to be a simulation relation from  $A_1$  to  $A_2$  if  $(q_1^0, q_2^0) \in R$ , and  $\forall (q_1, q_2) \in R, \delta_1(q_1, e) = q'_1$ , then  $\exists q'_2 \in Q_2$  such that  $\delta_2(q_2, e) = q'_2, (q'_1, q'_2) \in R$ .

If  $R$  is defined for all states and all events in  $A_1$ , then  $A_1$  is said to be similar to  $A_2$  (or  $A_2$  simulates  $A_1$ ), denoted by  $A_1 \prec A_2$  [27].

If  $A_1 \prec A_2, A_2 \prec A_1$ , with a symmetric relation, then  $A_1$  and  $A_2$  are said to be bisimilar (bisimulate each other), denoted by  $A_1 \cong A_2$  [28]. In general, bisimilarity implies languages equivalence but the converse does not necessarily hold [29].

In these works natural projection is used to obtain local tasks, since each agent has limited degree of sensing and actuation and hence it is provided with local information and functionalities: those events inside its local event set. Each agent may share some events with its neighbors to facilitate the cooperative control, using interactions between the connected agents. Natural projection is defined formally as follows.

*Definition 3: (Natural Projection on String)* Consider a global event set  $E$  and its local event sets  $E_i, i = 1, 2, \dots, n$ , with  $E = \bigcup_{i=1}^n E_i$ . Then, the natural projection  $p_i : E^* \rightarrow E_i^*$  is inductively

defined as  $p_i(\varepsilon) = \varepsilon$ , and  $\forall s \in E^*, e \in E : p_i(se) = \begin{cases} p_i(s)e & \text{if } e \in E_i; \\ p_i(s) & \text{otherwise.} \end{cases}$

Accordingly, inverse natural projection  $p_i^{-1} : E_i^* \rightarrow 2^{E^*}$  is defined on an string  $t \in E_i^*$  as  $p_i^{-1}(t) := \{s \in E^* | p_i(s) = t\}$ .

The natural projection is also defined on automata as  $P_i : A \rightarrow A$ , where,  $A$  is the set of finite automata and  $P_i(A_S)$  are obtained from  $A_S$  by replacing its events that belong to  $E \setminus E_i$  by  $\varepsilon$ -moves, and then, merging the  $\varepsilon$ -related states, forming equivalent classes defined as follows.

*Definition 4:* (Equivalent class of states, [30]) Consider an automaton  $A_S = (Q, q_0, E, \delta)$  and a local event set  $E_i \subseteq E$ . Then, the relation  $\sim_{E_i}$  is the equivalence relation on the set  $Q$  of states such that  $\delta(q, e) = q' \wedge e \notin E_i \Rightarrow q \sim_{E_i} q'$ , and  $[q]_{E_i}$  denotes the equivalence class of  $q$  defined on  $\sim_{E_i}$ . In this case,  $q$  and  $q'$  are said to be  $\varepsilon$ -related. The set of equivalent classes of states over  $\sim_{E_i}$ , is denoted by  $Q/\sim_{E_i}$  and defined as  $Q/\sim_{E_i} = \{[q]_{E_i} | q \in Q\}$ .

The natural projection is then formally defined on an automaton as follows.

*Definition 5:* (Natural Projection on Automaton) Consider an automaton  $A_S = (Q, q_0, E, \delta)$  and a local event set  $E_i \subseteq E$ . Then,  $P_i(A_S) = (Q_i = Q/\sim_{E_i}, [q_0]_{E_i}, E_i, \delta_i)$ , with  $\delta_i([q]_{E_i}, e) = [q']_{E_i}$  if there exist states  $q_1$  and  $q'_1$  such that  $q_1 \sim_{E_i} q$ ,  $q'_1 \sim_{E_i} q'$ , and  $\delta(q_1, e) = q'_1$ .

To investigate the interactions of transitions between automata, particularly between  $P_i(A_S)$ ,  $i = 1, \dots, n$ , the synchronized product of languages is defined as follows.

*Definition 6:* (Synchronized product of languages [10]) Consider a global event set  $E$  and local event sets  $E_i$ ,  $i = 1, \dots, n$ , such that  $E = \bigcup_{i=1}^n E_i$ . For a finite set of languages  $\{L_i \subset E_i^*\}_{i=1}^n$ , the synchronized product (language product) of  $\{L_i\}$ , denoted by  $\big|_{i=1}^n L_i$ , is defined as  $\big|_{i=1}^n L_i = \{s \in E^* | \forall i \in \{1, \dots, n\} : p_i(s) \in L_i\} = \bigcap_{i=1}^n p_i^{-1}(L_i)$ .

Then, capturing the interactions of agents, parallel composition (synchronized product) is used for two purposes: first to define the decomposition (as the parallel composition of local tasks should be equivalent to the original task), and second, to define the top-down cooperative control, such that the parallel composition local closed loop systems be equivalent to the global specification.

The parallel composition (synchronous product) is a way of modeling of interactions between agents as it allows local agents to transit on their own private events and restricts them to synchronize on the shared events, those events that are required for cooperation on common

actions or decision makings on orders or selections between events.

*Definition 7: (Parallel Composition)*

Let  $A_i = (Q_i, q_i^0, E_i, \delta_i)$ ,  $i = 1, 2$  be automata. The parallel composition (synchronous composition) of  $A_1$  and  $A_2$  is the automaton  $A_1 || A_2 = (Q = Q_1 \times Q_2, q_0 = (q_1^0, q_2^0), E = E_1 \cup E_2, \delta)$ ,

$$\text{with } \delta \text{ defined as } \forall (q_1, q_2) \in Q, e \in E: \delta((q_1, q_2), e) = \begin{cases} (\delta_1(q_1, e), \delta_2(q_2, e)), & \text{if } \begin{cases} \delta_1(q_1, e)!, \delta_2(q_2, e)! \\ e \in E_1 \cap E_2 \end{cases}; \\ (\delta_1(q_1, e), q_2), & \text{if } \delta_1(q_1, e)!, e \in E_1 \setminus E_2; \\ (q_1, \delta_2(q_2, e)), & \text{if } \delta_2(q_2, e)!, e \in E_2 \setminus E_1; \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

The parallel composition of  $A_i$ ,  $i = 1, 2, \dots, n$  is called parallel distributed system (or concurrent system), and is defined based on the associativity property of parallel composition [27] as

$$\bigparallel_{i=1}^n A_i = A_1 || \dots || A_n = A_n || (A_{n-1} || (\dots || (A_2 || A_1))).$$

The set of labels of local event sets containing an event  $e$  is called the set of locations of  $e$ , denoted by  $loc(e)$  and is defined as  $loc(e) = \{i \in \{1, \dots, n\} | e \in E_i\}$ .

In this sense, the decomposability of an automaton with respect to parallel composition and natural projections is defined as follows.

*Definition 8: (Automaton decomposability)* A task automaton  $A_S$  with the event set  $E$  and local event sets  $E_i$ ,  $i = 1, \dots, n$ ,  $E = \bigcup_{i=1}^n E_i$ , is said to be decomposable with respect to parallel composition and natural projections if  $\bigparallel_{i=1}^n P_i(A_S) \cong A_S$ .

In general, the task automaton can be nondeterministic. Decomposition of nondeterministic automaton however is very difficult to be characterized, due to interleaving of nondeterministic transitions between local task automata. For deterministic case, necessary and sufficient conditions for the decomposability of a deterministic task automaton  $A_S$  were proposed in [1] with respect to two cooperative agents and then generalized into an arbitrary finite number of agents, as follows.

*Lemma 1: (Corollary 1 in [2]):* A deterministic automaton  $A_S = \left(Q, q_0, E = \bigcup_{i=1}^n E_i, \delta\right)$  is decomposable with respect to parallel composition and natural projections  $P_i$ ,  $i = 1, \dots, n$  such that  $A_S \cong \bigparallel_{i=1}^n P_i(A_S)$  if and only if  $A_S$  satisfies the following decomposability conditions (DC):

- DC1:  $\forall e_1, e_2 \in E, q \in Q: [\delta(q, e_1)! \wedge \delta(q, e_2)!]$   
 $\Rightarrow [\exists E_i \in \{E_1, \dots, E_n\}, \{e_1, e_2\} \subseteq E_i] \vee [\delta(q, e_1 e_2)! \wedge \delta(q, e_2 e_1)!];$
- DC2:  $\forall e_1, e_2 \in E, q \in Q, s \in E^*: [\delta(q, e_1 e_2 s)! \vee \delta(q, e_2 e_1 s)!]$



- $\Rightarrow [\exists E_i \in \{E_1, \dots, E_n\}, \{e_1, e_2\} \subseteq E_i] \vee [\delta(q, e_1 e_2 s)! \wedge \delta(q, e_2 e_1 s)!];$
- *DC3*:  $\delta(q_0, \biguplus_{i=1}^n p_i(s_i))!, \forall \{s_1, \dots, s_n\} \in \tilde{L}(A_S), \exists s_i, s_j \in \{s_1, \dots, s_n\}, s_i \neq s_j$ , where,  $\tilde{L}(A_S) \subseteq L(A_S)$  is the largest subset of  $L(A_S)$  such that  $\forall s \in \tilde{L}(A_S), \exists s' \in \tilde{L}(A_S), \exists E_i, E_j \in \{E_1, \dots, E_n\}, i \neq j, p_{E_i \cap E_j}(s)$  and  $p_{E_i \cap E_j}(s')$  start with the same event, and
  - *DC4*:  $\forall i \in \{1, \dots, n\}, x, x_1, x_2 \in Q_i, x_1 \neq x_2, e \in E_i, t \in E_i^*, \delta_i(x, e) = x_1, \delta_i(x, e) = x_2: \delta_i(x_1, t)! \Leftrightarrow \delta_i(x_2, t)!$ .

Intuitively, the decomposability condition *DC1* means that for any decision on selection between two transitions there should exist at least one agent that is capable of the decision making, or the decision should not be important (both permutations in any order be legal). *DC2* says that for any decision on the order of two successive events before any string, either there should exist at least one agent capable of such decision making, or the decision should not be important, i.e., any order would be legal for occurrence of that string. The condition *DC3* means that the interleaving of strings from local task automata that synchronize on the same first appearing shared event, should not allow a string that is not allowed in the original task automaton. In other words, *DC3* is to ensure that an illegal behavior (an string that does not appear in  $A_S$ ) is not allowed by the team (does not appear in  $\biguplus_{i=1}^n P_i(A_S)$ ). In this condition,  $\biguplus_{i=1}^n p_i(s_i)$  is a language and stands for the interleaving or language product [10] of strings  $p_i(s_i)$ , defined as  $\biguplus_{i=1}^n p_i(s_i) = \bigcap_{i=1}^n p_i^{-1}(p_i(s_i))$ . The last condition, *DC4*, deals with the possible nondeterminisms in  $P_i(A_S)$  (Please note that here,  $A_S$  is considered to be deterministic, while  $P_i(A_S)$  can be nondeterministic, as it will be explained in Example 8). *DC4* ensures the determinism of bisimulation quotient of local task automata, in order to guarantee the symmetry of simulation relations between  $A_S$  and  $\biguplus_{i=1}^n P_i(A_S)$ . By providing this symmetry property, *DC4* guarantees that a legal behavior (an string in  $A_S$ ) is not disabled by the team (appears in  $\biguplus_{i=1}^n P_i(A_S)$ ).

In [2] it was also shown that for a decomposable task automaton, if local controllers exist such that each local closed loop system (parallel composition of local plant and local controller automata) satisfies its local task (bisimulates the corresponding local task automaton), then the controlled team of the agents will satisfy the global specification, as it is stated in the following lemma.

*Lemma 2:* (Theorem 2 in [2]): Consider a plant, represented by a parallel distributed system

$\prod_{i=1}^n A_{P_i}$ , with given local event sets  $E_i$ ,  $i = 1, \dots, n$ , and let the global specification is given by a deterministic task automaton  $A_S$ , with  $E = \bigcup_{i=1}^n E_i$ . If *DC1-DC4* are satisfied, then designing local controllers  $A_{C_i}$ , so that  $A_{C_i} \parallel A_{P_i} \cong P_i(A_S)$ ,  $i = 1, \dots, n$ , derives the global closed loop system to satisfy the global specification  $A_S$ , i.e.,  $\prod_{i=1}^n (A_{C_i} \parallel A_{P_i}) \cong A_S$ .

*Remark 1:* It is known that bisimulation implies language equivalence and that bisimulation of deterministic automata is reduced to their language equivalence. Now, one question is that whether for a deterministic task automaton its decomposability in the sense of bisimulation (stated in Lemma 1) is reduced to its decomposability in the sense of language equivalence ( $L(A_S) = L(\prod_{i=1}^n P_i(A_S))$ ) or its language separability ( $L(A_S) = \bigcap_{i=1}^n L(P_i(A_S))$ ). Furthermore, it is interesting to know whether the proposed top-down cooperative control, in Lemma 2, is reduced into a top-down approach in the sense of language equivalence. As it is illustrated in the Appendix, although in general, decomposability in the sense of bisimulation implies the decomposability in the sense of language equivalence, the reverse is not always true, in spite of determinism of automaton. For the top-down cooperative control, on the other hand, under the proposed decomposability conditions, the bisimulation-based approach is reduced to the language equivalence one, as the deterministic task automaton can be represented by its language.

To elaborate these remarks, we first highlight that the natural projection may impose emerging properties that do not exist in the original automaton. For example, local task automata may have some new strings that do not appear in the original automaton, i.e.,  $A_S$  does not necessarily simulates  $P_i(A_S)$ . Moreover, local task automata may become nondeterministic, even if the original task automaton is deterministic. The decomposability of  $A_S$ , however, concerns with bisimilarity of  $A_S$  and  $\prod_{i=1}^n P_i(A_S)$ , that may hold even if  $P_i(A_S) \not\sim A_S$ , or the local task automata are nondeterministic for some agents, as it is shown through examples in the Appendix.

### III. PROBLEM FORMULATION

In the previous section we recalled the conditions for task automaton decomposability to be used in top-down cooperative control. A natural follow-up question is that if after such decomposition, some of the events fail in some agents, then whether the global task automaton will still remain decomposable with respect to new set of events. And, if not, what are the conditions for preserving the decomposability. In order to address this problem, we first need to investigate the failure on events. In general, an event  $e$  can be either private ( $|loc(e)| = 1$ ) or

shared ( $|loc(e)| > 1$ ). Failure of private events fails the decomposability as it causes the failure in the whole team of agents. Failure on a shared event, on the other hand, may or may not lead to a global failure, depending on whether the failed event is redundant or not. When an event is a sensor reading; or actuator command, or it is sent to other agents with no other alternative links, then the failure on this event stops its global evolutions. In the following, we will introduce a class of failures that are investigated in this paper.

*Definition 9: (Event failure)* Consider an automaton  $A = (Q, q_0, E, \delta)$ . An event  $e \in E$  is said to be failed in  $A$  (or  $E$ ), if  $F(A) = P_\Sigma(A) = P_{E \setminus e}(A) = (Q, q_0, \Sigma = E \setminus e, \delta^F)$ , where,  $\Sigma$ ,  $\delta^F$  and  $F(A)$  denote the post-failure event set, post-failure transition relation and post-failure automaton, respectively. A set  $\bar{E} \subseteq E$  of events is then said to be failed in  $A$ , when for  $\forall e \in \bar{E}$ ,  $e$  is failed in  $A$ , i.e.,  $F(A) = P_\Sigma(A_i) = P_{E \setminus \bar{E}}(A) = (Q, q_0, \Sigma = E \setminus \bar{E}, \delta^F)$ .

Considering a parallel distributed plant  $A := \parallel_{i=1}^n A_i = (Z, z_0, E = \bigcup_{i=1}^n E_i, \delta_{||})$  with local agents  $A_i = (Q_i, q_0^i, E_i, \delta_i)$ ,  $i = 1, \dots, n$ . Failure of  $e$  in  $E_i$  is said to be passive in  $E_i$  (or  $A_i$ ) with respect to  $\parallel_{i=1}^n A_i$ , if  $E = \bigcup_{i=1}^n \Sigma_i$ . An event whose failure in  $A_i$  is a passive failure is called a passive event in  $A_i$ .

The notion of passivity, can be interpreted as communication redundancy as it is stated as follows.

*Remark 2:* To interpret the passivity more formally, let  $snd_e(i)$  and  $rcv_e(i)$  respectively denote the set of labels that  $A_i$  sends  $e$  to those agents and the set of labels that  $A_i$  receives  $e$  from their agents, defined as  $snd_e(i) = \{j \in \{1, \dots, n\} | A_i \text{ sends } e \text{ to } A_j\}$  and  $rcv_e(i) = \{j \in \{1, \dots, n\} | i \in snd_e(j)\}$ . Then, an event  $e$  is passive in  $A_i$  if  $rcv_e(i) \neq \emptyset$  (i.e., the  $i$ -th agent does not receive  $e$  from its own sensor/actuator readings, but from another agent), and  $\forall k \in snd_e(i)$ :  $\exists j \in \{1, \dots, n\} \setminus \{i, k\}, k \in snd_e(j)$  (i.e., if the  $i$ -th agent is a relay for transmission of  $e$ , for any receiver agent, there exist another agent to send  $e$ ). In this set-up a passive failure excludes the failed event  $e$  from the corresponding local event set  $E_i$  while it makes its respective transitions hidden in  $F(A_i)$ . Therefore, from definition of parallel composition, the transitions on other agents can contribute to form the global transitions in  $\parallel_{i=1}^n F(A_i)$ , since only in this way there will be no synchronization constraint on the rest of agents in  $\parallel_{i=1}^n F(A_i)$ .

Moreover, the definition of passivity implies that the passivity of failed events is a necessary condition for evolution of global transitions after failures, as it is stated in the following lemma.

*Lemma 3: (Global transitions after local failures)* Consider a parallel distributed plant  $A := \bigparallel_{i=1}^n A_i = (Z, z_0, E = \bigcup_{i=1}^n E_i, \delta_{\parallel})$  with local agents  $A_i = (Q_i, q_0^i, E_i, \delta_i)$ ,  $i = 1, \dots, n$ . If no global transitions in  $\bigparallel_{i=1}^n A_i$  are disabled in  $\bigparallel_{i=1}^n F(A_i)$  (i.e.,  $\forall z_1, z_2 \in Z, \forall e \in E, \delta_{\parallel}(z_1, e) = z_2$ , then  $\delta_{\parallel}^F(z_1, e) = z_2$ ), then all event failures are passive, i.e., the passivity of local event failures are necessary for preserving the global transitions.

*Proof:* See the proof in the Appendix. ■

The problem of decomposability under event failures is now defined as follows.

*Problem 1: (Decomposability under event failures)* Let a deterministic task automaton  $A_S = (Q, q_0, E = \bigcup_{i=1}^n E_i, \delta)$  is decomposable with respect to parallel composition and natural projections  $P_i$ ,  $i = 1, \dots, n$ . Then, does the global task automaton  $A_S$  remain decomposable in spite of failure of events  $\{a_{i,r}\}$ ,  $r \in \{1, \dots, n_i\}$  in local event sets  $E_i$ ,  $i \in \{1, \dots, n\}$ ? i.e., if  $A_S \cong \bigparallel_{i=1}^n P_i(A_S)$ , then does  $A_S \cong \bigparallel_{i=1}^n F(P_i(A_S))$  always hold true?, and if not, what are the conditions for such decomposability?

The next interesting question is the cooperative control under event failure, defined as

*Problem 2: (Cooperative tasking under event failure)* Consider a concurrent plant  $A_P := \bigparallel_{i=1}^n A_{P_i}$  and a decomposable deterministic task automaton  $A_S = (Q, q_0, E = \bigcup_{i=1}^n E_i, \delta) \cong \bigparallel_{i=1}^n P_i(A_S)$ , and suppose that local controller automata  $A_{C_i}$ ,  $i = 1, \dots, n$  exist such that each local closed loop system satisfies its corresponding local task, i.e.,  $A_{C_i} \parallel A_{P_i} \cong P_i(A_S)$ ,  $i = 1, \dots, n$ . Assume furthermore that  $\bar{E}_i = \{a_{i,r}\}$  fail in  $E_i$ ,  $r \in \{1, \dots, n_i\}$ . Then, does the team still can fulfill the global task, in spite of failures, without redesigning the controller automata, i.e.,  $\bigparallel_{i=1}^n F(A_{P_i} \parallel A_{C_i}) \cong A_S$ ?, and if not, what are the conditions to preserve the satisfaction of the global specification?

These problems will be addressed in the following two sections.

#### IV. TASK DECOMPOSABILITY UNDER EVENT FAILURES

According to definition of passivity, for any local event set  $E_i$ ,  $\Sigma_i$  excludes any passive failed events  $e$  from  $E_i$ , while the effect of this failure on  $P_i(A_S)$  is defined as the projection of  $A_S$  into  $E_i \setminus e$  (instead of  $E_i$ ), leading to  $P_{E_i \setminus e}(A_S)$ .

In this set up evolution of global transitions in  $\bigparallel_{i=1}^n F(P_i(A_S))$  relies on the passivity of failed events, as it is expected and stated in Lemma 3. The reason is that due to definition of parallel

composition, evolution of global transitions requires the failures to be passive, since passive failed events are excluded from the corresponding local event set and the local task automaton is projected to the rest of events. For non-passive failed events, on the other hand, since they are not received from other agents, and hence are not excluded from the local event set, but their transitions are stopped, then due to synchronization restriction in definition of parallel composition, the global transitions cannot evolve on them.

Consequently, as highlighted in Lemma 3, passivity of failed events is a necessary condition for the task automaton to remain decomposable after the failure.

Moreover, when all failed events are passive, due to definition of passivity, Problem 1 can be transformed into the standard decomposition problem to find the conditions under which  $A_S \cong \coprod_{i=1}^n P_{E_i \setminus \bar{E}_i}(A_S)$ . Accordingly, the conditions on the global task automaton to preserve the decomposability under event failures, are reduced into their respective decomposability conditions in Lemma 1, as the following lemmas.

*Lemma 4:* Consider a deterministic task automaton  $A_S = (Q, q_0, E = \bigcup_{i=1}^n E_i, \delta)$ . Assume that  $A_S$  is decomposable, i.e.,  $A_S \cong \coprod_{i=1}^n P_i(A_S)$ , and suppose that  $\bar{E}_i = \{a_{i,r}\}$  fail in  $E_i$ ,  $r \in \{1, \dots, n_i\}$ , and  $\bar{E}_i$  are passive for  $i \in \{1, \dots, n\}$ . Then, following two expressions are equivalent:

- 1) •  $EF1: \forall e_1, e_2 \in E, q \in Q: [\delta(q, e_1)! \wedge \delta(q, e_2)!]$   
 $\Rightarrow [\exists E_i \in \{E_1, \dots, E_n\}, \{e_1, e_2\} \subseteq E_i \setminus \bar{E}_i] \vee [\delta(q, e_1 e_2)! \wedge \delta(q, e_2 e_1)!];$   
 •  $EF2: \forall e_1, e_2 \in E, q \in Q, s \in E^*: [\delta(q, e_1 e_2 s)! \vee \delta(q, e_2 e_1 s)!]$   
 $\Rightarrow [\exists E_i \in \{E_1, \dots, E_n\}, \{e_1, e_2\} \subseteq E_i \setminus \bar{E}_i] \vee [\delta(q, e_1 e_2 s)! \wedge \delta(q, e_2 e_1 s)!].$
- 2) •  $DC1_\Sigma: \forall e_1, e_2 \in E, q \in Q: [\delta(q, e_1)! \wedge \delta(q, e_2)!]$   
 $\Rightarrow [\exists \Sigma_i \in \{\Sigma_1, \dots, \Sigma_n\}, \{e_1, e_2\} \subseteq \Sigma_i] \vee [\delta(q, e_1 e_2)! \wedge \delta(q, e_2 e_1)!];$   
 •  $DC2_\Sigma: \forall e_1, e_2 \in E, q \in Q, s \in E^*: [\delta(q, e_1 e_2 s)! \vee \delta(q, e_2 e_1 s)!]$   
 $\Rightarrow [\exists \Sigma_i \in \{\Sigma_1, \dots, \Sigma_n\}, \{e_1, e_2\} \subseteq \Sigma_i] \vee [\delta(q, e_1 e_2 s)! \wedge \delta(q, e_2 e_1 s)!].$

*Proof:* See the proof in the Appendix. ■

Lemma 4 gives the simplified versions of  $DC1$  and  $DC2$  after event failures, with respect to refined local event sets. Adopting the same  $DC3$  for the refined local event sets, it remains to represent a simplified version of  $DC4$  for the local task automata, after event failures. This condition is stated in the following lemma.

*Lemma 5:* Consider a deterministic task automaton  $A_S = (Q, q_0, E = \bigcup_{i=1}^n E_i, \delta)$ . Assume that  $A_S$  is decomposable, i.e.,  $A_S \cong \prod_{i=1}^n P_i(A_S)$ , and suppose that  $\bar{E}_i = \{a_{i,r}\}$  fail in  $E_i$ ,  $r \in \{1, \dots, n_i\}$ , and  $\bar{E}_i$  are passive for  $i \in \{1, \dots, n\}$ . Then, following two expressions are equivalent:

- *EF4:*  $\forall i \in \{1, \dots, n\}, x, x_1, x_2 \in Q_i, x_1 \neq x_2, e \in E_i \setminus \bar{E}_i, t_1 \in \bar{E}_i^*, t \in E_i^*, \delta_i(x, t_1 e) = x_1, \delta_i(x, e) = x_2: \delta_i(x_1, t)! \Leftrightarrow \delta_i(x_2, t)!$ .
- *DC4 $_{\Sigma}$ :*  $\forall i \in \{1, \dots, n\}, x, x_1, x_2 \in Q_i, x_1 \neq x_2, e \in \Sigma_i, t \in \Sigma_i^*, \delta_i^F(x, e) = x_1, \delta_i^F(x, e) = x_2: \delta_i^F(x_1, t)! \Leftrightarrow \delta_i^F(x_2, t)!$ . Where,  $\delta_i^F$  is the transition relation in  $F(P_i(A_S))$ .

*Proof:* See the proof in the Appendix. ■

*Remark 3:* *EF4* is the counterpart of *DC4* after the event failures, that handle newly possible nondeterminism in the local task automata. Any nondeterminism that is propagated from the local task automata of before the failure, is treated by *DC4* when  $A_S$  is decomposable.

Now, combination of Lemmas 1, 4, and 5 leads to the main result on decomposability under event failures as the following theorem.

*Theorem 1:* Consider a deterministic task automaton  $A_S = (Q, q_0, E = \bigcup_{i=1}^n E_i, \delta)$ . Assume that  $A_S$  is decomposable, i.e.,  $A_S \cong \prod_{i=1}^n P_i(A_S)$ , and furthermore, assume that  $\bar{E}_i = \{a_{i,r}\}$  fail in  $E_i$ ,  $r \in \{1, \dots, n_i\}$ , and  $\bar{E}_i$  are passive for  $i \in \{1, \dots, n\}$ . Then,  $A_S$  remains decomposable, in spite of event failures, i.e.,  $A_S \cong \prod_{i=1}^n F(P_i(A_S))$  if and only if

- *EF1:*  $\forall e_1, e_2 \in E, q \in Q: [\delta(q, e_1)! \wedge \delta(q, e_2)!] \Rightarrow [\exists E_i \in \{E_1, \dots, E_n\}, \{e_1, e_2\} \subseteq E_i \setminus \bar{E}_i] \vee [\delta(q, e_1 e_2)! \wedge \delta(q, e_2 e_1)!];$
- *EF2:*  $\forall e_1, e_2 \in E, q \in Q, s \in E^*: [\delta(q, e_1 e_2 s)! \vee \delta(q, e_2 e_1 s)!] \Rightarrow [\exists E_i \in \{E_1, \dots, E_n\}, \{e_1, e_2\} \subseteq E_i \setminus \bar{E}_i] \vee [\delta(q, e_1 e_2 s)! \wedge \delta(q, e_2 e_1 s)!];$
- *EF3:*  $\delta(q_0, \prod_{i=1}^n p_i(s_i))!, \forall \{s_1, \dots, s_n\} \in \tilde{L}(A_S), \exists s_i, s_j \in \{s_1, \dots, s_n\}, s_i \neq s_j$ , where  $\tilde{L}(A_S) \subseteq L(A_S)$  is the largest subset of  $L(A_S)$  such that  $\forall s \in \tilde{L}(A_S), \exists s' \in \tilde{L}(A_S), \exists \Sigma_i, \Sigma_j \in \{\Sigma_1, \dots, \Sigma_n\}, i \neq j, p_{\Sigma_i \cap \Sigma_j}(s)$  and  $p_{\Sigma_i \cap \Sigma_j}(s')$  start with the same event, and
- *EF4:*  $\forall i \in \{1, \dots, n\}, x, x_1, x_2 \in Q_i, x_1 \neq x_2, e \in E_i \setminus \bar{E}_i, t_1 \in \bar{E}_i^*, t \in E_i^*, \delta_i(x, t_1 e) = x_1, \delta_i(x, e) = x_2: \delta_i(x_1, t)! \Leftrightarrow \delta_i(x_2, t)!$ .

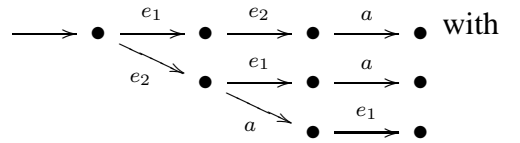
*Proof:* First, according to Lemma 3, passivity of  $\bar{E}_i$  is a necessary condition for preserving the decomposability. Now, providing the decomposability of  $A_S$  and passivity of all failed events, due to definition of passivity, it leads to  $\prod_{i=1}^n F(P_i(A_S)) \cong \prod_{i=1}^n P_{\Sigma_i}(A_S) = \prod_{i=1}^n P_{E_i \setminus \bar{E}_i}(A_S)$  that

based on Lemmas 1, 4 and 5, it is bisimilar to  $A_S$  if and only if  $EF1 - EF4$  hold true for the refined local event sets  $\{\Sigma_1, \dots, \Sigma_n\}$ . ■

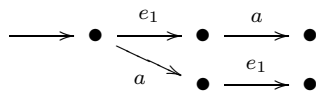
*Remark 4:*  $EF1$ - $EF4$  are respectively the decomposability conditions  $DC1$ - $DC4$ , after event failures with respect to parallel composition and natural projections into refined local event sets  $\Sigma_i = E_i \setminus \bar{E}_i$ ,  $i \in \{1, \dots, n\}$ , provided the passivity of  $\bar{E}_i$ ,  $i \in \{1, \dots, n\}$ . Condition  $EF1$  means that, after failure of some passive events, for any decision on selection between two transitions there should exist at least one agent that is capable of the decision making, or the decision should not be important (both permutations in any order be legal).  $EF2$  says that, after failure of some passive events, for any decision on the order of two successive events before any string, either there should exist at least one agent capable of such decision making, or the decision should not be important, i.e., any order would be legal for occurrence of that string. The condition  $EF3$  means that, after failure of some passive events, any interleaving of strings from local task automata that have the same first appearing shared event, should not allow a string that is not allowed in the original task automaton. In other words,  $EF3$  is to ensure that, after failure of some passive events, an illegal behavior (an string that does not appear in  $A_S$ ) is not allowed by the team (does not appear in  $\prod_{i=1}^n F(P_i(A_S))$ ). The last condition,  $EF4$ , ensures the determinism of bisimulation quotient of local task automaton, in order to guarantee the symmetry of simulation relations between  $A_S$  and  $\prod_{i=1}^n F(P_i(A_S))$ . By providing this symmetry property,  $EF4$  guarantees that, after the failures, a legal behavior (a string in  $A_S$ ) is not disabled by the team (appears in  $\prod_{i=1}^n F(P_i(A_S))$ ).

Following examples illustrate these conditions.

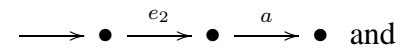
*Example 1:* This example illustrates the notion of passivity and shows a decomposable automaton that stays decomposable, when an even is failed passively in one of the local agents and  $EF1$ - $EF4$  are satisfied. Consider the automaton  $A_S$ :



local event sets  $E_1 = \{e_1, a\}$  and  $E_2 = \{e_2, a\}$ ,  $E_3 = \{a\}$  and communication pattern as  $\{1, 2\} \in \text{snd}_a(3)$ , and no other communication links. This automaton is decomposable, as the parallel composition of  $P_1(A_S) \cong$

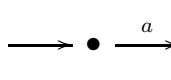


,  $P_2(A_S) \cong$

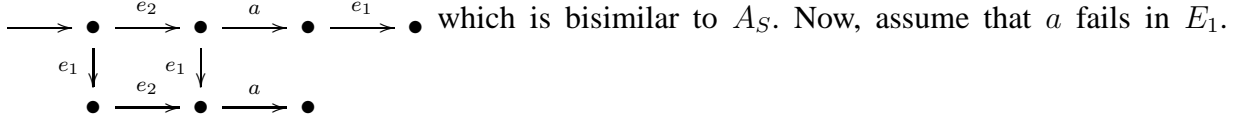


and

$P_3(A_S) \cong$

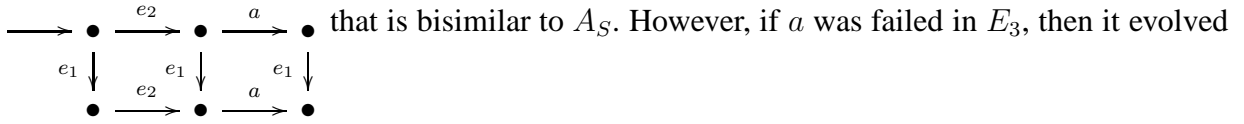


is  $\prod_{i=1}^3 P_i(A_S)$ :

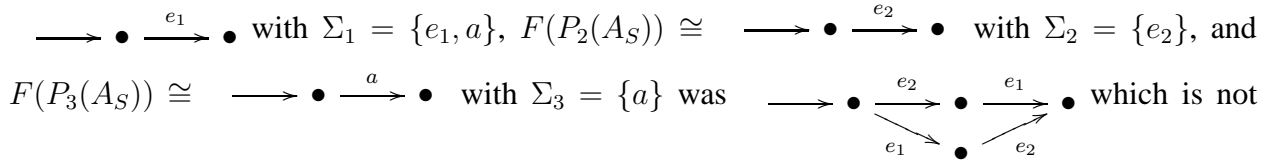


Then  $EF1$ - $EF4$  are satisfied (as  $\delta(q, e_2e_1a)! \wedge \delta(q, e_2ae_1)!$ , and hence,  $EF1$  and  $EF2$  hold true; after the failure, the interleavings on shared event  $a$  impose no illegal strings, and therefore,  $EF3$  is satisfied, and finally  $EF4$  is fulfilled since  $F(P_1(A_S)) \cong \longrightarrow \bullet \xrightarrow{e_1} \bullet$ ,  $F(P_2(A_S)) \cong \longrightarrow \bullet \xrightarrow{e_2} \bullet \xrightarrow{a} \bullet$  and  $F(P_3(A_S))$

$\cong \longrightarrow \bullet \xrightarrow{a} \bullet$  are all deterministic), and hence, the parallel composition of  $F(P_1(A_S))$  with  $\Sigma_1 = \{e_1\}$ ,  $F(P_2(A_S))$  with  $\Sigma_2 = \{e_2, a\}$ , and  $F(P_3(A_S))$  with  $\Sigma_3 = \{a\}$ , is  $\bigparallel_{i=1}^3 F(P_i(A_S))$ :



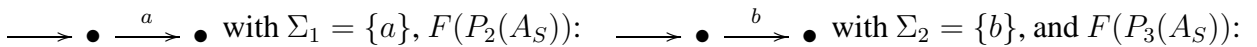
in none of the local task automata and  $\bigparallel_{i=1}^3 F(P_i(A_S)) \not\cong A_S$ , since  $E_3$  is a source for  $a$ . Similarly, failure of private events  $e_1$  and  $e_2$  in  $E_1$  and  $E_2$ , respectively, disables the global transitions on these events. As another example for non-passive failure, consider the communication pattern of  $1 \in \text{snd}_a(3)$ ,  $\{2, 3\} \subseteq \text{snd}_a(1)$ , while  $a$  fails in  $E_1$ . Then, the parallel composition of  $F(P_1(A_S))$ :



bisimilar to  $A_S$ . The reason is that in this case, in contrast to the first case,  $a$  was not excluded from  $\Sigma_1$ , while  $a$  was stopped in  $F(P_1(A_S))$ . This, due to the synchronization constraint in parallel composition, disabled the global transitions on  $a$ .

*Example 2:* This example shows a decomposable automaton that will no longer stay decomposable after a passive event failure, since  $EF1$  is not satisfied, although other three conditions,  $EF2$ ,  $EF3$  and  $EF4$ , are fulfilled. Consider the automaton  $A_S$ :  $\longrightarrow \bullet \xrightarrow{a} \bullet$  with local

event sets  $E_1 = \{a\}$ ,  $E_2 = \{b\}$ , and  $E_3 = \{a, b\}$  with  $3 \in \text{snd}_a(1)$  and  $3 \in \text{snd}_b(2)$ , and no other sending and receiving links. This automaton is decomposable, as the parallel composition of  $P_1(A_S)$ :  $\longrightarrow \bullet \xrightarrow{a} \bullet$ ,  $P_2(A_S)$ :  $\longrightarrow \bullet \xrightarrow{b} \bullet$  and  $P_3(A_S) \cong A_S$  bisimulates  $A_S$ . Now, suppose that  $a$  is failed in  $E_3$ . Then, the parallel composition of  $F(P_1(A_S))$ :





$\longrightarrow \bullet \xrightarrow{b} \bullet$  with  $\Sigma_3 = \{b\}$ , is  $\parallel_{i=1}^3 F(P_i(A_S))$ :  $\longrightarrow \bullet \xrightarrow{b} \bullet \xrightarrow{a} \bullet$  which is not

bisimilar to  $A_S$ . The reason is violation of *EF1*, as after the failure of  $a$  in  $E_3$ , neither there exists an agent that knows both events  $a$  and  $b$  to decide on the selection between them, nor both permutations are legal in  $A_S$ . If  $A_S$  was  $A_S$ :  $\longrightarrow \bullet \xrightarrow{a} \bullet \xrightarrow{b} \bullet$ , then, failure of  $a$  in  $E_3$  had no effect on decomposability of  $A_S$ .

*Example 3:* This example shows a decomposable automaton that will no longer stay decomposable after a passive failure, as *EF2* is not satisfied, although other three conditions, *EF1*, *EF3* and *EF4* are fulfilled. Consider the automaton  $A_S$ :  $\longrightarrow \bullet \xrightarrow{a} \bullet \xrightarrow{b} \bullet$  with local event sets  $E_1 = \{a\}$ ,  $E_2 = \{b\}$  and  $E_3 = \{a, b\}$ , with  $3 \in \text{snd}_a(1)$  and  $3 \in \text{snd}_b(2)$  with no other sending and receiving links. This automaton is decomposable, as the parallel composition of  $P_1(A_S)$ :  $\longrightarrow \bullet \xrightarrow{a} \bullet$ ,  $P_2(A_S)$ :  $\longrightarrow \bullet \xrightarrow{b} \bullet$  and  $P_3(A_S) \cong A_S$  bisimulates  $A_S$ . Now, suppose that  $a$  is failed in  $E_3$ . Then, the parallel composition of  $F(P_1(A_S))$ :

$\longrightarrow \bullet \xrightarrow{a} \bullet$  with  $\Sigma_1 = \{a\}$ ,  $F(P_2(A_S))$ :  $\longrightarrow \bullet \xrightarrow{b} \bullet$  with  $\Sigma_2 = \{b\}$ , and  $F(P_3(A_S))$ :  $\longrightarrow \bullet \xrightarrow{b} \bullet$  with  $\Sigma_3 = \{b\}$ , is  $\parallel_{i=1}^3 F(P_i(A_S))$ :  $\longrightarrow \bullet \xrightarrow{b} \bullet \xrightarrow{a} \bullet$  which is not

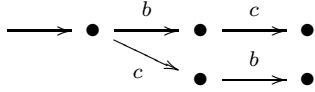
bisimilar to  $A_S$ . The reason is violation of *EF2*, as after the failure of  $a$  in  $E_3$ , neither there exists an agent that knows both events  $a$  and  $b$  to decide on the order of them, nor both orders are legal in  $A_S$ .

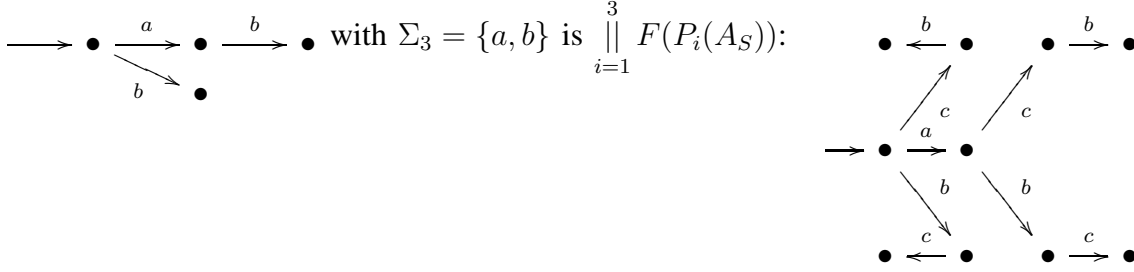
*Example 4:* This example illustrates a decomposable automaton that satisfies *EF1*, *EF2* and *EF4*, but it will not remain decomposable after a passive event failure, due to violation of *EF3*. Consider the automaton  $A_S$ :

$\longrightarrow \bullet \xrightarrow{a} \bullet \xrightarrow{b} \bullet \xrightarrow{c} \bullet$  with local event sets  $E_1 = \{a, b, c\}$ ,  $E_2 = \{b, c\}$  and  $E_3 = \{a, b\}$  and communication pattern  $1 \in \text{snd}_{\{b,c\}}(2)$ ,  $1 \in \text{snd}_a(3)$ ,  $3 \in \text{snd}_b(2)$ , with no other communication links.  $A_S$  is decomposable, as the parallel composition of  $P_1(A_S) \cong A_S$ ,  $P_2(A_S)$ :

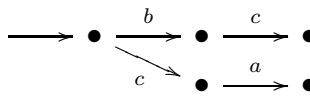
$\longrightarrow \bullet \xrightarrow{b} \bullet \xrightarrow{c} \bullet$ , and  $P_3(A_S)$ :  $\longrightarrow \bullet \xrightarrow{a} \bullet \xrightarrow{b} \bullet$  is bisimilar to  $A_S$ . Now, assume that  $b$  fails in  $E_1$ . Then, the parallel composition of  $F(P_1(A_S))$ :

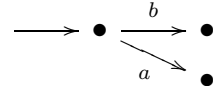
$\longrightarrow \bullet \xrightarrow{a} \bullet \xrightarrow{c} \bullet$  with

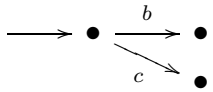
$\Sigma_1 = \{a, c\}$ ,  $F(P_2(A_S))$ :  with  $\Sigma_2 = \{b, c\}$  and  $F(P_3(A_S))$ :

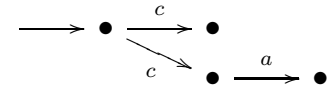


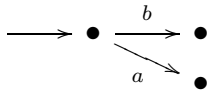
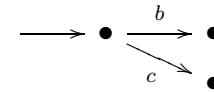
that is no longer bisimilar to  $A_S$  due to violation of  $EF3$  as it contains strings  $acb$  and  $bc$  that do not appear in  $A_S$ .

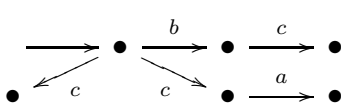
*Example 5:* This example shows a decomposable automaton that does not remain decomposable against a passive event failure, when it does not satisfy  $EF4$ , although it fulfils  $EF1$ ,  $EF2$  and  $EF3$ . Consider the automaton  $A_S$ :  with local event sets

$E_1 = \{a, b, c\}$ ,  $E_2 = \{a, b\}$  and  $E_3 = \{b, c\}$ , with communication structure  $1 \in \text{snd}_{\{a,b\}}(2)$ ,  $1 \in \text{snd}_c(3)$ ,  $3 \in \text{snd}_b(2)$ , with no other communication links. This automaton is decomposable, as the parallel composition of  $P_1(A_S) \cong A_S$ ,  $P_2(A_S)$ :  and  $P_3(A_S)$ :

 is bisimilar to  $A_S$ . Now, assume that  $b$  fails in  $E_1$ , then the parallel

composition of  $F(P_1(A_S))$ :  with  $\Sigma_1 = \{a, c\}$ ,  $F(P_2(A_S)) \cong$

 with  $\Sigma_2 = \{a, b\}$  and  $F(P_3(A_S)) \cong$   with  $\Sigma_3 =$

$\{b, c\}$  is  $\parallel_{i=1}^3 F(P_i(A_S))$ : 

that is no longer bisimilar to  $A_S$  due to violation of  $EF4$ , as there does not exist a deterministic automaton  $P'_1(A_S)$  such that  $P'_1(A_S) \cong F(P_1(A_S))$ .

## V. COOPERATIVE TASKING UNDER EVENT FAILURE

So far, we have presented the necessary and sufficient conditions for a decomposable task automaton to remain decomposable in spite of passive failures. Now, assume that the global task automaton is decomposable and local controllers have been designed in such a way that local

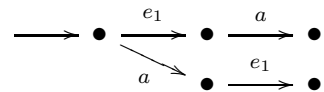
specifications are satisfied, and hence due to Lemma 2, the global specification is satisfied, by the team. Furthermore, assume that event failures occur on some shared events, but due to passivity of failed events and  $EF1$ - $EF4$ , the global task automaton remains decomposable. Then, the next question is Problem 2 to understand whether, the team is still able to achieve the global specification. Following result answers this question.

*Theorem 2:* Consider a concurrent plant  $A_P := \prod_{i=1}^n A_{P_i}$  and a deterministic task automaton  $A_S = (Q, q_0, E = \bigcup_{i=1}^n E_i, \delta)$  as the global specification. Assume that  $A_S$  is decomposable, i.e.,  $A_S \cong \prod_{i=1}^n P_i(A_S)$ , and suppose that local controller automata  $A_{C_i}$ ,  $i = 1, \dots, n$  have been designed such that each local closed loop system satisfies its corresponding local task, i.e.,  $A_{C_i} \parallel A_{P_i} \cong P_i(A_S)$ ,  $i = 1, \dots, n$ . Assume furthermore that  $\bar{E}_i = \{a_{i,r}\}$  fail in  $E_i$ ,  $r \in \{1, \dots, n_i\}$ ,  $\bar{E}_i$  are passive for  $i \in \{1, \dots, n\}$ , and  $A_S$  satisfies  $EF1$ - $EF4$ . Then, the team can still achieve its global specification, i.e.,  $\prod_{i=1}^n F(A_{P_i} \parallel A_{C_i}) \cong A_S$ .

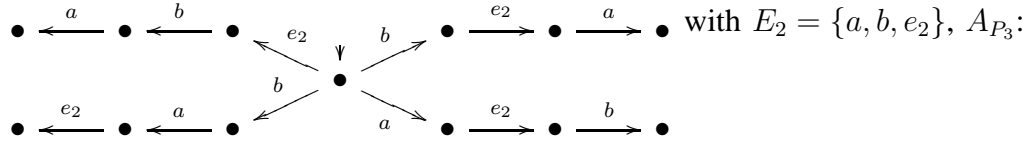
*Proof:* Firstly, decomposability of  $A_S$  and  $A_{C_i} \parallel A_{P_i} \cong P_i(A_S)$ ,  $i = 1, \dots, n$ , due to Lemma 2, implies that  $\prod_{i=1}^n (A_{P_i} \parallel A_{C_i}) \cong A_S$ , i.e., the global specification is satisfied by the team. Moreover, the global specification remains satisfied, in spite of event failures, if  $\bar{E}_i$  are passive for  $i \in \{1, \dots, n\}$ , and  $A_S$  satisfies  $EF1$ - $EF4$ , since  $\prod_{i=1}^n F(A_{P_i} \parallel A_{C_i}) \cong \prod_{i=1}^n P_{E_i \setminus \bar{E}_i}(A_{P_i} \parallel A_{C_i}) \cong \prod_{i=1}^n P_{E_i \setminus \bar{E}_i}(P_i(A_S)) \cong \prod_{i=1}^n F(P_i(A_S)) \cong \prod_{i=1}^n P_i(A_S) \cong A_S$ . In this expression, the first and the third bisimilarities come from passivity of  $\bar{E}_i$ ,  $i \in \{1, \dots, n\}$ , and the second bisimilarity is followed from  $A_{C_i} \parallel A_{P_i} \cong P_i(A_S)$ ,  $i = 1, \dots, n$ , definition of natural projection and from the fact  $(A_1 \cong A_2) \wedge (A_3 \cong A_4) \Rightarrow (A_1 \parallel A_3 \cong A_2 \parallel A_4)$  (Lemma 6 in [1]). The fourth equivalence is implied from passivity of  $\bar{E}_i$ ,  $i = 1, \dots, n$  and  $EF1$ - $EF4$ , and finally, the last bisimilarity is due to the decomposability assumption of  $A_S$ . ■

*Remark 5:* The significance of Theorem 2 is that under passivity condition and  $EF1$ - $EF4$ , although local task automata may change after the failure (i.e.,  $F(P_i(A_S)) \not\cong P_i(A_S)$ ), the team of agents can satisfy the global specification, as  $\prod_{i=1}^n F(A_{P_i} \parallel A_{C_i}) \cong \prod_{i=1}^n F(P_i(A_S)) \cong \prod_{i=1}^n P_i(A_S) \cong A_S$ .

*Example 6:* This example illustrates a specification for a team of three agents that is globally satisfied and remains satisfied in spite of passive event failures, provided  $EF1$ - $EF4$ . Consider a concurrent plant  $A_P := \prod_{i=1}^3 A_{P_i}$  with local plants  $A_{P_1}$ :

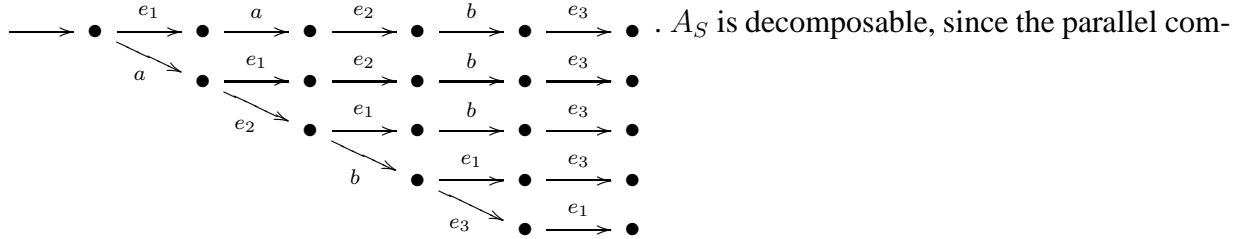


$E_1 = \{a, e_1\}$ ,  $A_{P_2}$ :



with  $E_2 = \{a, b, e_2\}$ ,  $A_{P_3}$ :

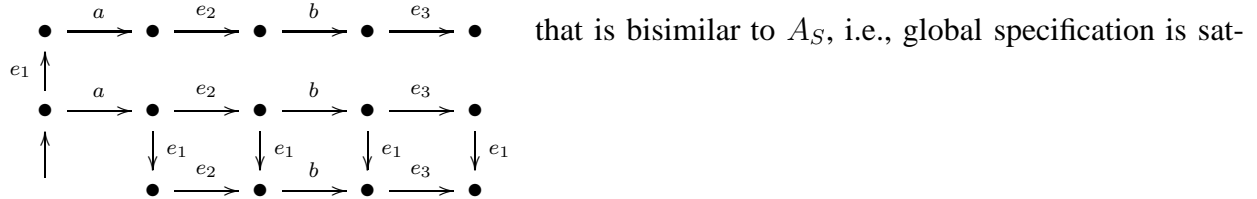
having communication pattern  $1 \in \text{send}_a(2)$ ,  $3 \in \text{send}_b(2)$ , and no more communication links. Assume that the global specification is given as  $A_S$ :



position of  $P_1(A_S) \cong$  ,  $P_2(A_S) \cong$

and  $P_3(A_S) \cong$  is bisimilar to

$A_S$ . Now, taking local controller as  $A_{C_i} := P_i(A_S)$ ,  $i = 1, 2, 3$  results in  $A_{P_i} \parallel A_{C_i} \cong P_i(A_S)$ ,  $i = 1, 2, 3$  and  $\bigparallel_{i=1}^3 (A_{P_i} \parallel A_{C_i}) \cong$

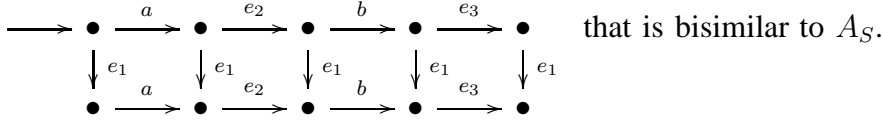


isfied by designing local controllers  $A_{C_i}$  to satisfy local satisfactions  $P_i(A_S)$ .

Now, suppose that  $a$  fails in  $E_1$ . Since  $a$  is passive in  $E_1$  and  $A_S$  satisfies  $EF1$ - $EF4$  (since  $\delta(q_0, e_1 a e_2 b e_3)! \wedge \delta(q_0, a e_1 e_2 b e_3)!$  in  $A_S$ , and hence  $EF1$  and  $EF2$  are satisfied;  $\Sigma_1 = \{e_1\}$ ,  $\Sigma_2 = \{a, b, e_2\}$ ,  $\Sigma_3 = \{b, e_3\}$  with the only shared events  $b \in \Sigma_2 \cap \Sigma_3$ , and the corresponding interleaving between  $F(P_2(A_S)) \cong P_2(A_S)$  and  $F(P_3(A_S)) \cong P_3(A_S)$  is  $a e_2 b e_3$  that appears in  $A_S$ , with all permutations with  $e_1$  from  $F(P_1(A_S))$ , and hence,  $EF3$  is satisfied, and finally,  $EF4$  is fulfilled since  $F(P_1(A_S))$ ,  $F(P_2(A_S))$  and  $F(P_3(A_S))$  are respectively bisimilar to automata

, and that all are deterministic. Therefore, according to Theorem 1,  $\bigparallel_{i=1}^3 F(P_i(A_S)) \cong A_S$ .

Moreover, since the failed event  $a$  is passive in  $E_1$  and  $A_S$  satisfies  $EF1$ - $EF4$ , as Theorem 2, the global specification remains satisfied after failure, as  $\prod_{i=1}^3 F(A_{P_i} || A_{C_i}) \cong \prod_{i=1}^3 F(P_i(A_S)) \cong$



## VI. SPECIAL CASE: MORE INSIGHT INTO 2-AGENT CASE

This part provides a closer look into the two agent case and illustrated the notion of global decision making after the event failures.

First, following lemma presents some properties on a 2-agent system that experiences passive failures. The properties will be then used to provide a deeper insight on the global decision making of the team on successive and adjacent transitions, in spite of passive failures.

*Lemma 6:* Consider a deterministic task automaton  $A_S = (Q, q_0, E = E_1 \cup E_2, \delta)$  and assume that  $A_S$  is decomposable with respect to parallel composition and natural projections  $P_i, i = 1, 2$ , and furthermore assume that  $\bar{E}_i = \{a_{i,r}\}, r \in \{1, \dots, n_i\}$  fail in  $E_i, i \in \{1, 2\}$ . If  $\bar{E}_i, i \in \{1, 2\}$  are passive, then  $\forall i \in \{1, 2\}$

- 1)  $\bar{E}_1 \cap \bar{E}_2 = \emptyset$ ;
- 2)  $\bar{E}_1, \bar{E}_2 \in E_1 \cap E_2$ ;
- 3)  $\Sigma_1 \setminus \Sigma_2 = (E_1 \setminus E_2) \cup \bar{E}_2$  and  $\Sigma_2 \setminus \Sigma_1 = (E_2 \setminus E_1) \cup \bar{E}_1$ .

Now, following lemma represents the conditions for maintaining the capability of a team of two cooperative agents for global decision making on the orders and selections of transitions in the global task automaton, after passive event failures.

*Lemma 7:* Consider a deterministic task automaton  $A_S = (Q, q_0, E = E_1 \cup E_2, \delta)$ . Assume that  $A_S$  is decomposable, i.e.,  $A_S \cong P_1(A_S) || P_2(A_S)$ , and furthermore, assume that  $\bar{E}_i = \{a_{i,r}\}$  fail in  $E_i, r \in \{1, \dots, n_i\}$ , and  $\bar{E}_i$  are passive for  $i \in \{1, 2\}$ . Then, the following two expressions are equivalent:

- ( $EF1$  and  $EF2$ ):  $\forall (e_1, e_2) \in \{(E_1 \setminus E_2, \bar{E}_1), (E_2 \setminus E_1, \bar{E}_2), (\bar{E}_1, \bar{E}_2)\}, q \in Q, s \in E^*$ :

$$[\delta(q, e_1)! \wedge \delta(q, e_2)!] \Rightarrow [\delta(q, e_1 e_2)! \wedge \delta(q, e_2 e_1)!] \quad (1)$$

$$\delta(q, e_1 e_2 s)! \Leftrightarrow \delta(q, e_2 e_1 s)! \quad (2)$$

- $(DC1_\Sigma \text{ and } DC2_\Sigma): \forall e_1 \in \Sigma_1 \setminus \Sigma_2, e_2 \in \Sigma_2 \setminus \Sigma_1, q \in Q, s \in E^*:$

$$[\delta(q, e_1)! \wedge \delta(q, e_2)!] \Rightarrow [\delta(q, e_1 e_2)! \wedge \delta(q, e_2 e_1)!]$$

$$\delta(q, e_1 e_2 s)! \Leftrightarrow \delta(q, e_2 e_1 s)!.$$

*Proof:* See the proof in the Appendix. ■

*Remark 6:*  $EF1$  and  $EF2$  represent the decomposability conditions  $DC1$  and  $DC2$  after failure, i.e., for the refined local event sets  $\Sigma_1$  and  $\Sigma_2$ . They say that after the failure, any decision on the switch or the order between two events that cannot be accomplished by at least one of the agents (neither  $\{e_1, e_2\} \subseteq \Sigma_1$ , nor  $\{e_1, e_2\} \subseteq \Sigma_2$ ), then the decision should not be important (both orders should be legal). This is a good insight on validity of  $DC1$  and  $DC2$  after failure of passive events as it is illustrated in Figure 1, based on the properties in Lemma 6.

From Lemma 6,  $(\Sigma_1 \setminus \Sigma_2) \times (\Sigma_2 \setminus \Sigma_1)$  is the union of four spaces:  $(E_1 \setminus E_2) \times (E_2 \setminus E_1)$ ;  $(E_1 \setminus E_2) \times (\bar{E}_1)$ ;  $(\bar{E}_2) \times (E_2 \setminus E_1)$ , and  $(\bar{E}_1) \times (\bar{E}_2)$  (see Figure 1(a) – (d)). Note that due to Lemma 6,  $\bar{E}_1 \cap \bar{E}_2 = \emptyset$ .

Now, according to Lemma 8 in the Appendix, for any pair of events from  $(E_1 \setminus E_2) \times (E_2 \setminus E_1)$  (shown in Figure 1 – (a)), (1) and (2) are true as  $A_S$  is decomposable, before the failure. Moreover, (1) and (2) are also true for the pair of events from other three spaces of  $(\Sigma_1 \setminus \Sigma_2) \times (\Sigma_2 \setminus \Sigma_1)$ , due to  $EF1$  and  $EF2$  as it is illustrated as follows.

- Figure 1 – (b) shows  $(E_1 \setminus E_2) \times (\bar{E}_1)$ : any pair of events from this space contains in  $E_1$ , before the failure, but, contains in neither of  $E_1$  and  $E_2$  after the failure;
- Figure 1 – (c) depicts  $(\bar{E}_2) \times (E_2 \setminus E_1)$ : any pair of events from this space contains in  $E_2$ , before the failure, but, belongs to neither of  $E_1$  and  $E_2$  after the failure;
- Figure 1 – (d) illustrates  $(\bar{E}_1) \times (\bar{E}_2)$ : any pair of events from this space contains in both  $E_1$  and  $E_2$ , before the failure, but, contains in none of them after the failure.

Therefore, since after the failure, for any pair of events from these three spaces, no agent can be responsible for decision making on switch/order between them (no local event set contains both events), then such decisions should not be important as it stated in  $EF1$  and  $EF2$ .

Another implication of this result is that when the system is comprised of only two agents and one of those agent is failed, while all of its events are passive, then the task automaton

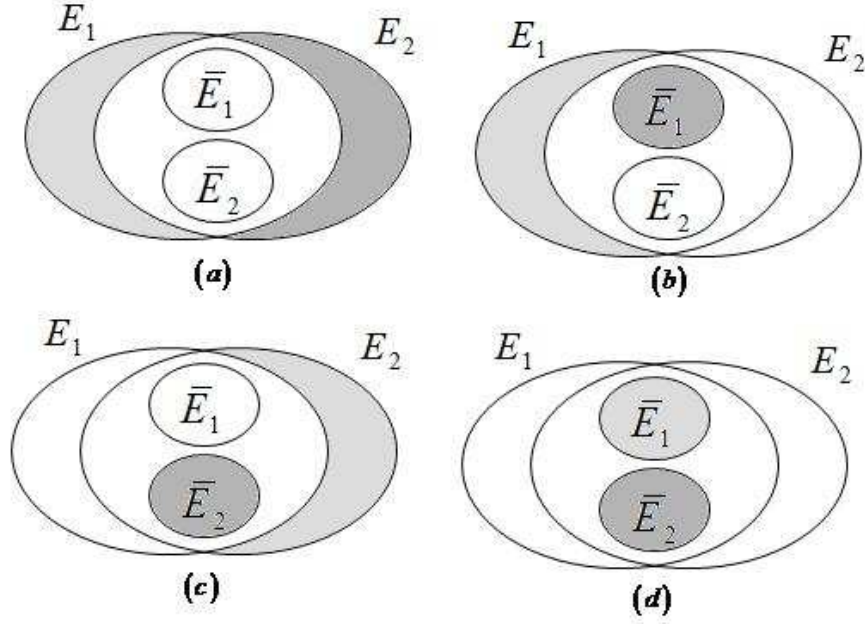


Fig. 1. Illustration of  $(\Sigma_1 \setminus \Sigma_2) \times (\Sigma_2 \setminus \Sigma_1) = [(E_1 \setminus E_2) \times (E_2 \setminus E_1)] \cup [(E_1 \setminus E_2) \times (\bar{E}_1)] \cup [(\bar{E}_2) \times (E_2 \setminus E_1)] \cup [(\bar{E}_1) \times (\bar{E}_2)]$ , (a):  $(E_1 \setminus E_2) \times (E_2 \setminus E_1)$ ; (b):  $(E_1 \setminus E_2) \times (\bar{E}_1)$ ; (c):  $(\bar{E}_2) \times (E_2 \setminus E_1)$ , and (d):  $(\bar{E}_1) \times (\bar{E}_2)$ .

remains decomposable as

*Corollary 1:* Consider a deterministic task automaton  $A_S = (Q, q_0, E = E_1 \cup E_2, \delta)$ . Assume that  $A_S$  is decomposable, i.e.,  $A_S \cong P_1(A_S) || P_2(A_S)$ . Assume furthermore that  $E_1$  entirely fails, i.e.,  $\bar{E}_1 = E_1$ . Then,  $A_S \cong \bigparallel_{i=1}^2 F(P_i(A_S))$  if and only if  $\bar{E}_1$  is passive.

*Proof: Sufficiency:* Since  $\bar{E}_1 = E_1$ , from definition of passivity, Lemma 6 and  $E = E_1 \cup E_2$ , it follows that  $E_1 \subseteq E_2 = E$  and  $E_1 \setminus E_2 = \bar{E}_2 = \emptyset$ , and hence,  $EF1$  and  $EF2$  hold true, due to Lemma 7. Moreover, since  $\Sigma_1 = E_1 \setminus \bar{E}_1 = \emptyset$ , then  $\Sigma_1 \setminus \Sigma_2 = \Sigma_1 = \emptyset$ , that makes  $EF3$  always true. Finally, by Lemma 3,  $F(P_1(A_S))$  with  $\Sigma_1 = \emptyset$  merges into its initial state, with no nondeterminism, and  $F(P_2(A_S))$  with  $\Sigma_2 = E$  is bisimilar to  $A_S$  which is deterministic, therefore,  $EF4$  is satisfied, as well. This implies that when  $\bar{E}_1 = E_1$ , the passivity of  $\bar{E}_1$  leads to  $A_S \cong F(P_1(A_S)) || F(P_2(A_S))$ .

*Necessity:* The necessity is proven by contradiction. Suppose that  $\bar{E}_1 = E_1$  and  $A_S \cong F(P_1(A_S)) || F(P_2(A_S))$ , but  $\exists e \in \bar{E}_1$ ,  $e$  is not passive in  $E_1$ . Then, from Lemma 3, it follows that transitions on  $e$  cannot evolve in  $F(P_1(A_S)) || F(P_2(A_S))$ , due to synchronization constraint in parallel composition, and hence,  $A_S \not\cong F(P_1(A_S)) || F(P_2(A_S))$  which is a contradiction. ■

## VII. CONCLUSIONS

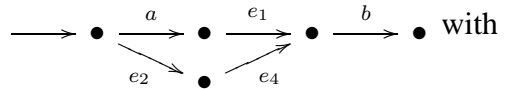
This paper proposed a formal method to investigate whether a decentralized bisimilarity control design remains valid, under failure of some events in multi-agent systems. This work is a continuation of [1], [2], in which necessary and sufficient condition was given for task automaton decomposition and the satisfaction of global specification was guaranteed up on satisfaction of local specifications. This work then defines a new notion of passivity under which it is possible to transform the decentralized cooperative control problem under event failures into the standard decomposability problem in [1], [2] and identifies necessary and sufficient conditions to still guarantee the supervised concurrent plant to satisfy the global specification, in spite of event failures. The passivity of the failed events is turned to be a necessary condition for the task automaton to remain decomposable, and it is found to reflect the failure of redundant communication links. It is then proven that a decomposable task automaton remains decomposable and satisfied after some passive failures if and only if after the failures, the team of agents maintain the capability on collective decision making on the orders and selections of transitions and preserve the collective perceiving of the task such that the parallel composition of local task automata neither allow an illegal behavior (a string that is not in the global task automaton), nor disallow a legal behavior ( a string from the global task automaton).

This result is of practical importance as it provides a sense of fault-tolerance to the task decomposition and top-down cooperative control of multi-agent systems, under event failures.

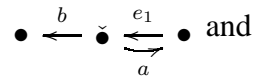
## VIII. APPENDIX

### A. Examples for Remark 1

*Example 7:* Following example shows an automaton that does not simulate its natural projections, yet is decomposable. Consider an automaton  $A_S$ :



the event set  $E = E_1 \cup E_2$  and local event sets  $E_1 = \{a, b, e_1\}$ ,  $E_2 = \{a, b, e_2, e_4\}$ . In this example,  $A_S$  is decomposable, since it bisimulates the parallel composition of  $P_1(A_S)$ :



$P_2(A_S)$ : , although  $P_1(A_S) \not\sim A_S$  (since the string  $b$  appears in

$P_1(A_S)$ , but not in  $A_S$ ), and  $P_2(A_S) \not\sim A_S$  (since the string  $ab$  appears in  $P_2(A_S)$ , but not in  $A_S$ ).



As mentioned in Remark 1, another emergent property is that natural projection of local task automata may lead to nondeterminism of  $P_i(A_S)$ , leading to nondeterminism of  $\bigsqcup_{i=1}^n P_i(A_S)$ . The decomposability of  $A_S$  again concerns with bisimilarity of  $A_S$  and  $\bigsqcup_{i=1}^n P_i(A_S)$ , that may happen even if there exist some nondeterministic  $P_i(A_S)$ , as it is elaborated in the following example.

*Example 8:* Consider the automaton  $A_S$ :

with  $E = E_1 \cup E_2$ ,  $E_1 = \{a, e_1\}$ ,  $E_2 = \{a, e_2\}$ .  $A_S$  is decomposable, as the parallel composition of  $P_1(A_S)$ :

and  $P_2(A_S)$ :

the deterministic automaton  $P_2(A_S)'$ :

Therefore, a deterministic task automaton  $A_S$  may have nondeterministic natural projections, and consequently, its  $\bigsqcup_{i=1}^n P_i(A_S)$  may become nondeterministic. As a result, determinism of  $A_S$  does not reduce its decomposability in the sense of bisimulation into its decomposability in the sense of language equivalence (synthesis modulo language equivalence [9]), due to possibility of nondeterminism of  $\bigsqcup_{i=1}^n P_i(A_S)$ , as it is further illustrated in the following example.

*Example 9:* Consider the task automaton  $A_S$ :

with  $E_1 = \{a, b, e_1\}$ ,  $E_2 = \{a, b\}$ , leading to  $P_1(A_S)$ :

$P_2(A_S)$ :

$P_1(A_S) \parallel P_2(A_S)$ :

which is not bisimilar to  $A_S$ . In this example  $A_S$  is deterministic,  $L(\bigsqcup_{i=1}^n P_i(A_S)) = L(A_S)$ ; however,  $\bigsqcup_{i=1}^n P_i(A_S) \not\cong A_S$ .

This example also shows that determinism of  $A_S$  also does not reduce its decomposability in the sense of bisimulation into the separability of its language ([10]), as  $\bigsqcup_{i=1}^n P_i(A_S) \not\cong A_S$ , although  $A_S$  is deterministic and its language is separable ( $L(A_S) = \bigcap_{i=1}^n L(P_i(A_S))$ ).

Therefore, in general for a deterministic task automaton  $\bigsqcup_{i=1}^n P_i(A_S) \cong A_S$  is not reduced into

$L(A_S) = \bigcup_{i=1}^n L(P_i(A_S))$ . But, under the determinism of bisimulation quotient of all local task automata (*DC4*), bisimulation-based decomposability is reduced to language-based decomposability and the top-down design based on bisimulation, is reduced to language-based top-down design, such that the entire closed loop system (the parallel composition of local closed loop systems) bisimulates (or equivalently is language equivalent to) the global task automaton. In case of *DC4*, the other three conditions (*DC1-DC3*) can be used to characterize the language separability.

### B. Proof for Lemma 3

Firstly, in order to allow the global transitions, the failed event  $a$  in  $E_i$  has to be received from other agents not from its own sensors and actuator readings, otherwise, no local transitions on  $a$  evolve in either of  $F(A_i)$  or  $\bigcup_{i=1}^n F(A_i)$  (since other agents receive  $a$  from  $A_i$ ). Therefore, the failed events have to necessarily be shared events ( $\text{loc}(a) > 1$ ), and that after the failure of  $a$  in  $A_i$ ,  $a$  is excluded from  $E_i$ , i.e.,  $\Sigma_i = E_i \setminus a$ , as  $a$  is not received to  $A_i$  from other agents. Moreover, due to Definition 7, exclusion of  $a$  from  $E_i$  allows global transitions on  $a$  with no synchronization restriction from  $F(A_i)$ . Finally, the transitions on failed event  $a$  has to be replaced with  $\varepsilon$ -moves, in order to allow transitions after  $a$  in  $A_i$ , i.e.,  $\forall x_1, x_2 \in Q_i$ ,  $\delta_i(x_1, a) = x_2$ , then  $\delta_i^F([x_1]_{\Sigma_i}, a) = [x_2]_{\Sigma_i}$ ,  $[x_1]_{\Sigma_i} = [x_2]_{\Sigma_i}$  and  $F(A_i) = P_{E_i \setminus a}(A_i)$  (otherwise a transition of  $\delta_i^F(\delta_i^F(x, a), e)$  will be disabled due to stopping of execution of  $\delta_i^F(x, a)$ ). It should be noted that, if there are no traditions after  $\delta_i(x, a)$  (i.e.,  $\forall e \in E_i: \neg \delta_i(\delta_i(x, a), e)!$ ), then stopping of  $\delta_i(x, a)$  is identical to replacing this transition with an  $\varepsilon$ -move. These collectively mean that preserving of global transitions in  $\bigcup_{i=1}^n F(A_i)$  requires then local failures to be passive.

### C. Proof for Lemma 4

Passivity of all  $\bar{E}_i$ ,  $i \in \{1, \dots, n\}$ , due to definition of passivity, leads to  $\Sigma_i = E_i \setminus \bar{E}_i \subseteq E_i$ , and hence, the expression  $[\exists E_i \in \{E_1, \dots, E_n\}, \{e_1, e_2\} \subseteq E_i]$  in the antecedent of *DC1* and *DC2* leads to  $[\exists \Sigma_i \in \{\Sigma_1, \dots, \Sigma_n\}, \{e_1, e_2\} \subseteq \Sigma_i]$ , replacing  $E_i$  with  $\Sigma_i = E_i \setminus \bar{E}_i$ .

### D. Proof for Lemma 5

Any nondeterminism in  $F(P_i(A_S))$  appears either due to nondeterminism from  $P_i(A_S)$  or newly formed nondeterminism because of replacing of passive events by  $\varepsilon$ .

In the first case, from decomposability of  $A_S$ ,  $DC4$  says that for any  $x, x_1, x_2 \in Q_i, e \in E_i \setminus \bar{E}_i, t \in E_i^*, x_1 \neq x_2, \delta_i(x, e) = x_1, \delta_i(x, e) = x_2: \delta_i(x_1, t)! \Leftrightarrow \delta_i(x_2, t)!, \text{ i.e., } \delta_i^F([x]_{\Sigma_i}, e) = [x_1]_{\Sigma_i}, \delta_i^F([x]_{\Sigma_i}, e) = [x_2]_{\Sigma_i}: \delta_i^F([x_1]_{\Sigma_i}, p_{\Sigma_i}(t))! \Leftrightarrow \delta_i^F([x_2]_{\Sigma_i}, p_{\Sigma_i}(t))!, \text{ which is } DC4 \text{ for } F(P_i(A_S)), \text{ with refined local event set } \Sigma_i.$

For the second case, any newly appeared nondeterminism is induced by transitions from the original local task automaton, in the following form.  $\exists i \in \{1, \dots, n\}, x, x_1, x_2 \in Q_i, t_1 \in \bar{E}_i^*, e \in E_i \setminus \bar{E}_i, t \in E_i^*, x_1 \neq x_2, \delta_i(x, t_1 e) = x_1, \delta_i(x, e) = x_2$  then  $[x]_{\Sigma_i} = [\delta_i^F([x]_{\Sigma_i}, t_1)]_{\Sigma_i}$ , and hence,  $EF4$  becomes  $\delta_i^F([x]_{\Sigma_i}, e) = [x_1]_{\Sigma_i}, \delta_i^F([x]_{\Sigma_i}, e) = [x_2]_{\Sigma_i}: \delta_i^F([x_1]_{\Sigma_i}, p_{\Sigma_i}(t))! \Leftrightarrow \delta_i^F([x_2]_{\Sigma_i}, p_{\Sigma_i}(t))!$ , which is again equivalent to  $DC4$  for  $F(P_i(A_S))$ .

### E. Proof for Lemma 6

The first item is proven based on the fact that if  $\exists e \in \bar{E}_1 \cap \bar{E}_2$ , then  $snd_e(1) = \emptyset \wedge snd_e(2) = \emptyset$  which is impossible, due to Remark 2 that requires  $snd_e(i) = \emptyset \wedge rec_e(i) \neq \emptyset$  for an event  $e$  to be passive in  $E_i \in \{E_1, E_2\}$ , in two agent case.

The second item, comes from passivity of  $\bar{E}_1$  and  $\bar{E}_2$  that implies that  $\forall e \in \bar{E}_i, i = 1, 2, snd_e(i) = \emptyset \wedge rcv_e(i) \neq \emptyset$ , and hence  $loc(e) > 1$  which means  $e \in E_j, j \in \{1, 2\} \setminus \{i\}$ , i.e.,  $e \in E_1 \cap E_2$ .

For the last item, from the second item and  $\bar{E}_1 \cap \bar{E}_2 = \emptyset$  we respectively have  $\bar{E}_1, \bar{E}_2 \subseteq E_1 \cap E_2$  and  $\bar{E}_1 \subseteq \bar{E}_2', \bar{E}_2 \subseteq \bar{E}_1'$  (In this proof, prime operation stands for the set complements, where the  $E_1 \cup E_2$  is considered as the universal set). Consequently,  $\Sigma_1 \setminus \Sigma_2 = (E_1 \setminus \bar{E}_1) \setminus (E_2 \setminus \bar{E}_2) = (E_1 \cap \bar{E}_1') \cap (E_2 \cap \bar{E}_2')' = (E_1 \cap \bar{E}_1') \cap (E_2' \cup \bar{E}_2) = [(E_1 \cap \bar{E}_1') \cap E_2'] \cup [(E_1 \cap \bar{E}_1') \cap \bar{E}_2] = [E_1 \cap (\bar{E}_1 \cup E_2)'] \cup [(E_1 \cap \bar{E}_2) \cap \bar{E}_1'] = (E_1 \cap E_2') \cup (\bar{E}_2 \cap \bar{E}_1') = (E_1 \setminus E_2) \cup \bar{E}_2$ . Similarly,  $\Sigma_2 \setminus \Sigma_1 = (E_2 \setminus E_1) \cup \bar{E}_1$ .

### F. Proof for Lemma 7

To prove this lemma, firstly, the decomposability result for two agents is recalled as

**Lemma 8:** (Theorem 1 in [1]) A deterministic automaton  $A_S = (Q, q_0, E = E_1 \cup E_2, \delta)$  is decomposable with respect to parallel composition and natural projections  $P_i, i = 1, 2$ , such that  $A_S \cong P_1(A_S) || P_2(A_S)$  if and only if it satisfies the following decomposability conditions:  $\forall e_1 \in E_1 \setminus E_2, e_2 \in E_2 \setminus E_1, q \in Q, s \in E^*$ ,

- $DC1: [\delta(q, e_1)! \wedge \delta(q, e_2)!] \Rightarrow [\delta(q, e_1 e_2)! \wedge \delta(q, e_2 e_1)!];$

- $DC2$ :  $\delta(q, e_1 e_2 s)! \Leftrightarrow \delta(q, e_2 e_1 s)!$ ;
- $DC3$ :  $\forall s, s' \in E^*$ , sharing the same first appearing common event  $a \in E_1 \cap E_2$ ,  $s \neq s'$ ,  $q \in Q$ :  $\delta(q, s)! \wedge \delta(q, s')! \Rightarrow \delta(q, p_1(s)|p_2(s'))! \wedge \delta(q, p_1(s')|p_2(s))!$ , and
- $DC4$ :  $\forall i \in \{1, 2\}$ ,  $x, x_1, x_2 \in Q_i$ ,  $x_1 \neq x_2$ ,  $e \in E_i$ ,  $t \in E_i^*$ ,  $\delta_i(x, e) = x_1$ ,  $\delta_i(x, e) = x_2$ :  $\delta_i(x_1, t)! \Leftrightarrow \delta_i(x_2, t)!$ .

Now, in order to prove the equivalence of two cases in lemma 7, one needs to prove that the set  $\{\bar{E}_1 \times E_1 \setminus E_2, \bar{E}_2 \times E_2 \setminus E_1, \bar{E}_1 \times \bar{E}_2\}$  in  $EF1$  and  $EF2$  is equal to the set  $\{(\Sigma_1 \setminus \Sigma_2) \times (\Sigma_2 \setminus \Sigma_1)\}$  in  $DC1_\Sigma$  and  $DC1_\Sigma$  (decomposability conditions  $DC1$  and  $DC2$  with respect to  $\Sigma_1$  and  $\Sigma_2$ ).

From lemma 6,  $e_1 \in \Sigma_1 \setminus \Sigma_2$ ,  $e_2 \in \Sigma_2 \setminus \Sigma_1$  is equivalent to  $e_1 \in (E_1 \setminus E_2) \cup \bar{E}_2$ ,  $e_2 \in (E_2 \setminus E_1) \cup \bar{E}_1$  which means that  $e_1 \in E_1 \setminus E_2 \vee e_1 \in \bar{E}_2$  and  $e_2 \in E_2 \setminus E_1 \vee e_2 \in \bar{E}_1$ , leading to four possible cases:  $(e_1 \in E_1 \setminus E_2 \wedge e_2 \in E_2 \setminus E_1)$ ,  $(e_1 \in E_1 \setminus E_2 \wedge e_2 \in \bar{E}_1)$ ,  $(e_1 \in \bar{E}_2 \wedge e_2 \in E_2 \setminus E_1)$  or  $(e_1 \in \bar{E}_2 \wedge e_2 \in \bar{E}_1)$ .

Now, Lemma 7 is proven as follows. For the first case, since decomposability of  $A_S$  implies  $DC1$  and  $DC2$ , then,  $\forall e_1 \in E_1 \setminus E_2, e_2 \in E_2 \setminus E_1, q \in Q, s \in E^*$ : (1) and (2) hold true. For the second, third and fourth cases, i.e., when  $(e_1 \in E_1 \setminus E_2 \wedge e_2 \in \bar{E}_1)$ ,  $(e_1 \in \bar{E}_2 \wedge e_2 \in E_2 \setminus E_1)$  or  $(e_1 \in \bar{E}_2 \wedge e_2 \in \bar{E}_1)$ , then (1) and (2) are guarantee by  $EF1$  and  $EF2$ . Therefore, provided the decomposability of  $A_S$ ,  $EF1$  and  $EF2$ , (1) and (2) become true for all  $e_1 \in \Sigma_1 \setminus \Sigma_2, e_2 \in \Sigma_2 \setminus \Sigma_1$ . This means that  $EF1$  and  $EF2$  are respectively equivalent to  $DC1$  and  $DC2$  after failures (for  $\Sigma_1$  and  $\Sigma_2$ ).

## REFERENCES

- [1] M. Karimadini and H. Lin, "Guaranteed global performance through local coordinations," *Automatica*, vol. In Press, Corrected Proof, 2011. [Online]. Available: <http://www.sciencedirect.com/science/article/B6V21-529M7SH-5/2/19d8df66047d3ddc475af4b1b79ef8d2>
- [2] —, "Necessary and sufficient conditions for task automaton decomposition," *submitted to IEEE Transactions on Automatic Control*, vol. 2011, 2011.
- [3] P. U. Lima and L. M. Custodio, *Multi-Robot Systems, Book Series Studies in Computational Intelligence, Book Innovations in Robot, Mobility and Control*. Berlin: Springer Berlin / Heidelberg, 2005, vol. 8.
- [4] G. E. Fainekos, A. Girard, H. Kress-Gazit, and G. J. Pappas, "Temporal logic motion planning for dynamic robots," *Automatica*, vol. 45, no. 2, pp. 343–352, 2009.
- [5] Z. Ji, Z. Wang, H. Lin, and Z. Wang, "Brief paper: Interconnection topologies for multi-agent coordination under leader-follower framework," *Automatica*, vol. 45, no. 12, pp. 2857–2863, 2009.
- [6] V. R. Lesser, "Cooperative multiagent systems: A personal view of the state of the art," *IEEE Transactions on Knowledge and Data Engineering*, vol. 11, pp. 133–142, 1999.

- [7] J. Choi, S. Oh, and R. Horowitz, "Distributed learning and cooperative control for multi-agent systems," *Automatica*, vol. 45, no. 12, pp. 2802–2814, 2009.
- [8] E. Semsar-Kazerooni and K. Khorasani, "Multi-agent team cooperation: A game theory approach," *Automatica*, vol. 45, no. 10, pp. 2205–2213, 2009.
- [9] M. Mukund, *From global specifications to distributed implementations*, in B. Caillaud, P. Darondeau, L. Lavagno (Eds.), *Synthesis and Control of Discrete Event Systems*, Kluwer. Berlin: Springer Berlin / Heidelberg, 2002.
- [10] Y. Willner and M. Heymann, "Supervisory control of concurrent discrete-event systems," *International Journal of Control*, vol. 54, pp. 1143–1169, 1991.
- [11] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis, "Diagnosability of discrete-event systems," *Automatic Control, IEEE Transactions on*, vol. 40, no. 9, pp. 1555 –1575, Sept. 1995.
- [12] S. Takai and T. Ushio, "Reliable decentralized supervisory control of discrete event systems," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 30, no. 5, pp. 661 –667, Oct. 2000.
- [13] F. Liu and H. Lin, "Reliable supervisory control for general architecture of decentralized discrete event systems," *Automatica*, vol. 46, no. 9, pp. 1510 – 1516, 2010. [Online]. Available: <http://www.sciencedirect.com/science/article/B6V21-50FHMXXJ-3/2/49559880c5e773d537069b2f5ac6e318>
- [14] F. Lin, "Robust and adaptive supervisory control of discrete event systems," *Automatic Control, IEEE Transactions on*, vol. 38, no. 12, pp. 1848 –1852, Dec. 1993.
- [15] H. Darabi, M. Jafari, and A. Buczak, "A control switching theory for supervisory control of discrete event systems," *Robotics and Automation, IEEE Transactions on*, vol. 19, no. 1, pp. 131 – 137, Feb. 2003.
- [16] K. Rohloff, "Sensor failure tolerant supervisory control," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, 2005, pp. 3493 – 3498.
- [17] S. Lafortune and F. Lin, "On tolerable and desirable behaviors in supervisory control of discrete event systems," in *Decision and Control, 1990., Proceedings of the 29th IEEE Conference on*, Dec. 1990, pp. 3434 –3439 vol.6.
- [18] R. M. Jensen, *A survey of formal methods for intelligent swarmsDES Controller Synthesis and Fault Tolerant Control: A Survey of Recent Advances, Tech. Rep. TR-2003-40*. IT Univ. of Copenhagen, Copenhagen, Denmark: NASA Goddard Space Flight Center, 2003.
- [19] Q. Wen, R. Kumar, J. Huang, and H. Liu, "A framework for fault-tolerant control of discrete event systems," *Automatic Control, IEEE Transactions on*, vol. 53, no. 8, pp. 1839 –1849, 2008.
- [20] Y. Brave and M. Heymann, "On stabilization of discrete event processes," *Int. J. Control*, vol. 51, no. 5, pp. 1101–1117, 1990.
- [21] C. M. Özveren, A. S. Willsky, and P. J. Antsaklis, "Stability and stabilizability of discrete event dynamic systems," *J. ACM*, vol. 38, no. 3, pp. 729–751, 1991.
- [22] R. Kumar, V. Garg, Marcus, and S. I. Marcus, "Language stability and stabilizability of discrete event dynamical systems," *SIAM Journal of Control and Optimization*, vol. 31, no. 5, pp. 1294–1320, 1993.
- [23] Y. Willner and M. Heymann, "Language convergence in controlled discrete-event systems," *Automatic Control, IEEE Transactions on*, vol. 40, no. 4, pp. 616 –627, Apr. 1995.
- [24] A. Saboori and S. Zad, "Fault recovery in discrete event systems," in *Computational Intelligence Methods and Applications, 2005 ICSC Congress on*, 0 2005.
- [25] M. Karimadini and H. Lin, "Reliable task decomposability for cooperative multi-agent systems," in *submitted to the 30th Chinese Control Conference, CCC2011*, 2011.

- [26] R. Kumar and V. K. Garg, *Modeling and Control of Logical Discrete Event Systems*. Norwell, MA, USA: Kluwer Academic Publishers, 1999.
- [27] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. USA: Springer, 2008.
- [28] C. Zhou, R. Kumar, and S. Jiang, “Control of nondeterministic discrete-event systems for bisimulation equivalence,” *Automatic Control, IEEE Transactions on*, vol. 51, no. 5, pp. 754 – 765, may 2006.
- [29] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas, “Discrete abstractions of hybrid systems,” *Proceedings of the IEEE*, vol. 88, no. 7, pp. 971 –984, jul 2000.
- [30] R. Morin, “Decompositions of asynchronous systems,” in *CONCUR '98: Proceedings of the 9th International Conference on Concurrency Theory*. London, UK: Springer-Verlag, 1998, pp. 549–564.